Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for exporting a specification and description language (SDL) software model to different operating systems, the method comprising:

providing an SDL software model;

providing an SDL porting layer, the SDL porting layer converting the SDL software model to an operating environment wherein

the operating environment is common to all the different operating systems; and

providing a plurality of operating system abstraction layers, each abstraction layer designed to abstract the operating environment to at least one targeted operating system;

wherein the plurality of abstraction layers utilize a naming convention to specify which modules are operating system (OS) dependent and which are OS independent; and wherein the abstraction layer comprises a plurality of OS constructs, the plurality of OS constructs further comprising:

a Thread, said thread including an independent path of execution;

a Process, said process including an independent path of execution with its own protected address space;

a Thread Group, said Thread Group including a grouping of threads, managed collectively to synchronize their execution;

a Mutex, said Mutex including a Thread synchronization element providing mutual exclusion to shared resources; and

an Event, said Event including a Thread synchronization element, allowing threads to coordinate execution.

2. (Original) The method of claim 1 wherein the at least one targeted operating system is a single operating system.

3. (Original) The method of claim 1 wherein the at least one targeted operating system is two operating systems and the method for exporting a software model in a wireless device, a first of the two operating systems is a system operating system and a second of the two operating systems is a communication operating system.

4. (Original) The method of claim 3 wherein the system operating system operates on an advanced reduced instruction set processor (RISC) and the communication operating system operates on a digital signal processor (DSP).

5. (Original) The method of claim 3 wherein a communication module facilitates communication between the RISC and DSP.

6. (Original) The method of claim 5 wherein the communication module has an associated shared memory for use in performing operations of code derived from the software model.

7. (Original) The method of claim 1 wherein the at least one target operating system is a plurality of operating systems.

8. (Original) The method of claim 1 wherein the operating environment operates independently of processor boundaries.

9. (Original) The method of claim 7 wherein the operating system abstraction layer defines the processor boundaries and facilitates communication across the processor boundaries.

10. (Currently Amended) A wireless communication device comprising:

at least one system processor and at least one communication processor;

a communication module to facilitate communication between each system and communication processor;

a shared memory associated with the communication module;

each system processor and communication processor having an associated operating system, the operating system performing code generated from an SDL software model, the SDL software model being ported to an operating environment wherein the operating environment is the result of an SDL porting layer converting an SDL software model to the operating environment, providing an operating environment, the operating environment common to all the different operating systems, an operating system abstraction layer abstracts the operating environment to each associated operating system;

wherein the abstraction layer utilizes a naming convention to specify which modules are operating system (OS) dependent and which are OS independent; and wherein the abstraction layer comprises a plurality of OS constructs, the plurality of OS constructs further comprising:

a Thread, said thread including an independent path of execution;

a Process, said process including an independent path of execution with its own protected address space;

a Thread Group, said Thread Group including a grouping of threads, managed collectively to synchronize their execution;

a Mutex, said Mutex including a Thread synchronization element providing mutual exclusion to shared resources; and

an Event, said Event including a Thread synchronization element, allowing threads to coordinate execution.

11. (Original) The wireless communication device of claim 10 wherein the wireless communication device is a wireless transmit/receive unit.

12. (Original) The wireless communication device of claim 11 wherein the at least one system processor is a advanced reduced instruction set processor and the communication processor is a digital signal processor.

13. (Original) The wireless communication device of claim 10 wherein the operating environment operates independently of processor boundaries.

14. (Original) The wireless communication device of claim 13 wherein the operating system abstraction layer defines the processor boundaries and facilitates communication across the processor boundaries.

15-30. (Canceled).

31. (Currently Amended) A wireless transmit/receive unit (WTRU) comprising a processor further comprising:

an operating system abstraction layer comprising:

an interface with an operating environment, wherein the operating environment is the result of an SDL porting layer converting an SDL software model to the operating environment, the operating environment operating independent of underlying operating systems;

an operating system independent module for performing operations that are not related to a target operating system;

an operating system dependent module for performing operations that are related to the target operating system; and

an interface with the target operating system;

wherein the abstraction layer utilizes a naming convention to specify which modules are operating system (OS) dependent and which are OS independent; and wherein the abstraction layer comprises a plurality of OS constructs, the plurality of OS constructs further comprising:

a Thread, said thread including an independent path of execution;

a Process, said process including an independent path of execution with its own protected address space;

a Thread Group, said Thread Group including a grouping of threads, managed collectively to synchronize their execution;

a Mutex, said Mutex including a Thread synchronization element providing mutual exclusion to shared resources; and

an Event, said Event including a Thread synchronization element, allowing threads to coordinate execution.

32. (Currently Amended) A method for abstracting an operating environment to a plurality of operating systems, the method comprising:

providing an operating environment wherein the operating environment is the result of an SDL porting layer converting an SDL software model to the operating environment, the operating environment common to all the different operating systems; and

providing a plurality of operating system abstraction layers, each abstraction layer designed to abstract the operating environment to at least one targeted operating system;

wherein each abstraction layer utilizes a naming convention to specify which modules are operating system (OS) dependent and which are OS independent; and

wherein each abstraction layer comprises a plurality of OS constructs, the plurality of OS constructs further comprising:

a Thread, said thread including an independent path of execution;

a Process, said process including an independent path of execution with its own protected address space;

a Thread Group, said Thread Group including a grouping of threads, managed collectively to synchronize their execution;

a Mutex, said Mutex including a Thread synchronization element providing mutual exclusion to shared resources; and

an Event, said Event including a Thread synchronization element, allowing threads to coordinate execution.

33. (Original) The method of claim 32 wherein each abstraction layer has a same operating system dependent module and a different operating system independent module.

34. (Currently Amended) A wireless communication device comprising:

at least one system processor and at least one communication processor;

a communication module to facilitate communication between each system and communication processor;

a shared memory associated with the communication module;

each system processor and communication processor having an associated operating system, the operating system performing code from an operating system abstraction layer, the abstraction layer interfacing with the operating environment wherein the operating environment is the result of an SDL porting layer converting an SDL software model to the operating environment and having an operating system independent module for performing operations that are not related to a target operating system and an operating system dependent module for performing operations that are related to the target operating system;

wherein the abstraction layer utilizes a naming convention to specify which modules are operating system (OS) dependent and which are OS independent; and wherein the abstraction layer comprises a plurality of OS constructs, the plurality of OS constructs further comprising:

a Thread, said thread including an independent path of execution;

a Process, said process including an independent path of execution with its own protected address space;

a Thread Group, said Thread Group including a grouping of threads, managed collectively to synchronize their execution;

a Mutex, said Mutex including a Thread synchronization element providing mutual exclusion to shared resources; and

an Event, said Event including a Thread synchronization element, allowing threads to coordinate execution.

35. (Original) The wireless communication device of claim 34 wherein the wireless communication device is a wireless transmit/receive unit.

36-47. (Canceled).